SLAM Demo

EDAA – G06

João Martins
Henrique Ribeiro – João Costa
Tiago Duarte

Simultaneous Location And Mapping

- **Goal** – map an environment navigated by and autonomous vehicle, while simultaneously locating it in the map;
- **Challenges:**
  - No access to pre-existing maps or external devices;
  - Focus on sub-aquatic SLAM ⇒ difficult access and extra data noise;
- **Datasets** – the group will have access to sonar data measured by CRAS.

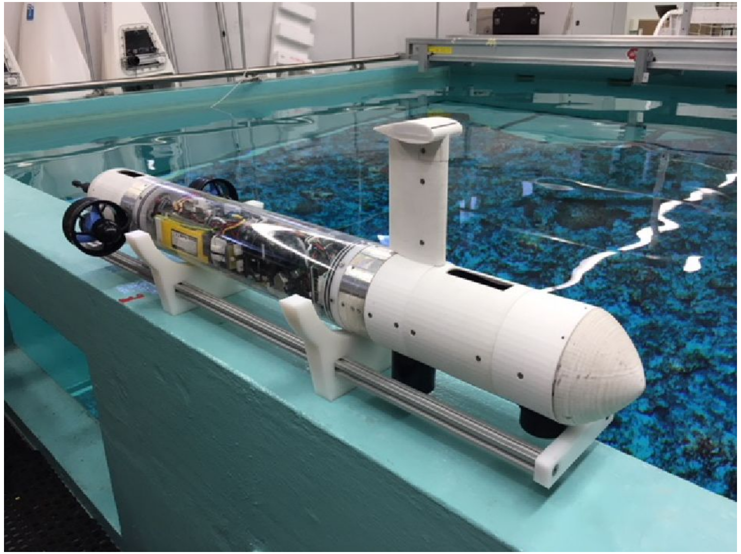
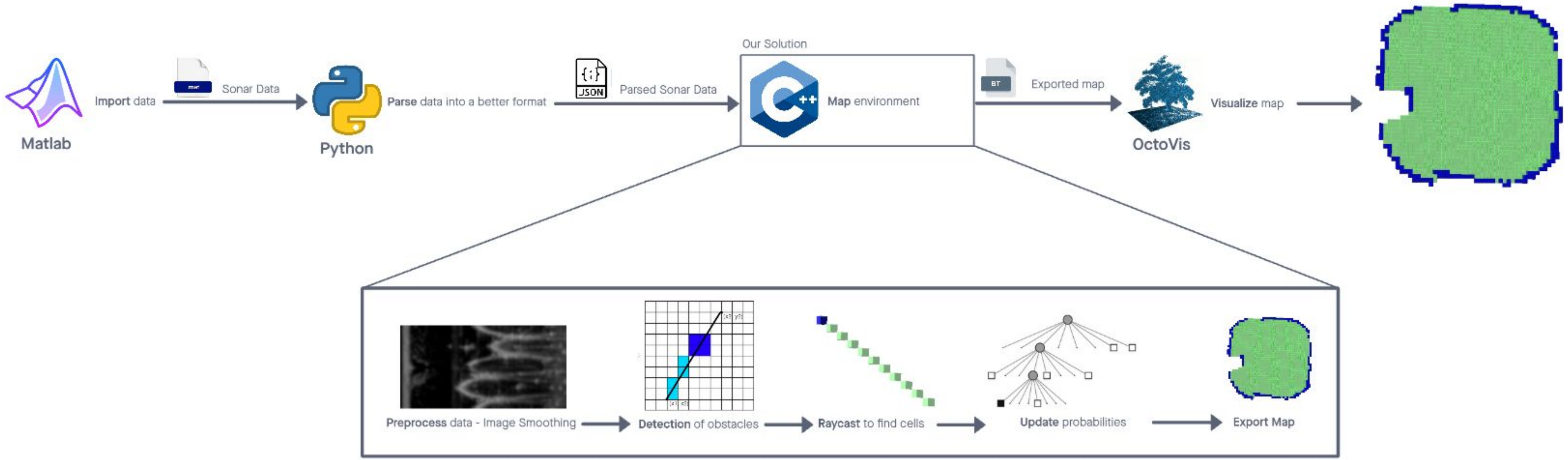


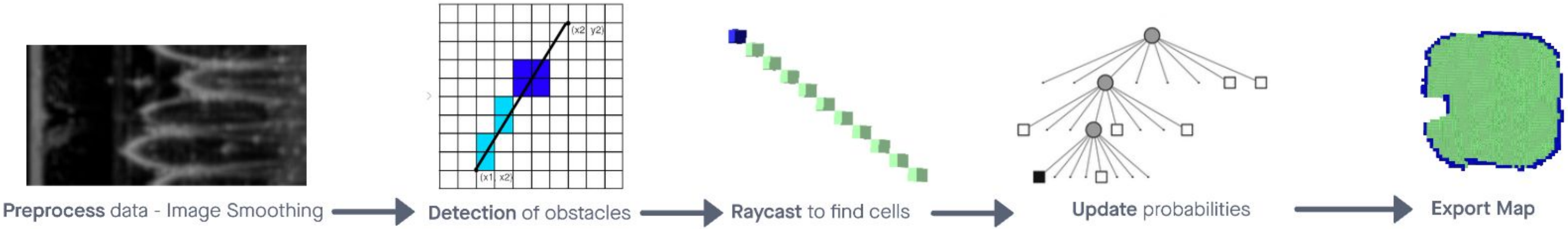
Fig 1. UAV used to collect the datasets.

Pipeline



Matlab
Import data
Sonar Data
Python
Parse data into a better format
JSON
Parsed Sonar Data
Our Solution
Map environment
BT
Exported map
OctoVis
Visualize map
Preprocess data - Image Smoothing
Detection of obstacles
Raycast to find cells
Update probabilities
Export Map

Pipeline



Preprocess data - Image Smoothing
Detection of obstacles
Raycast to find cells
Update probabilities
Export Map

Data Preprocessing - Image Smoothing

- Use blurring to reduce noise
- Improves drastically edge detection
- Applied to polar coordinates

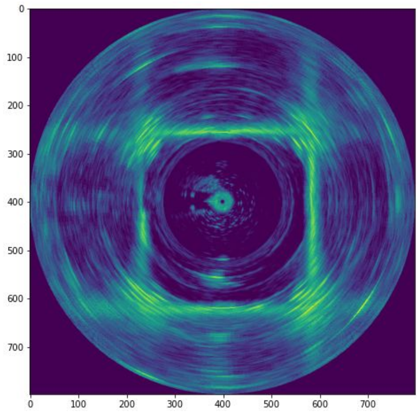


Fig 27. Dataset representation in Cartesian coordinates.

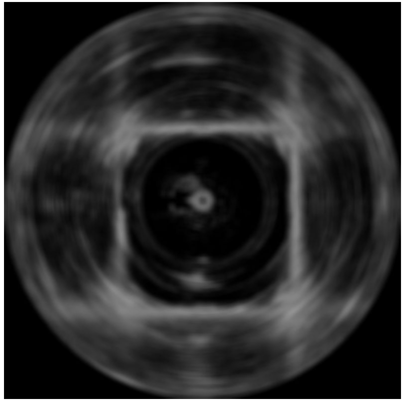


Fig 28. Gaussian filter

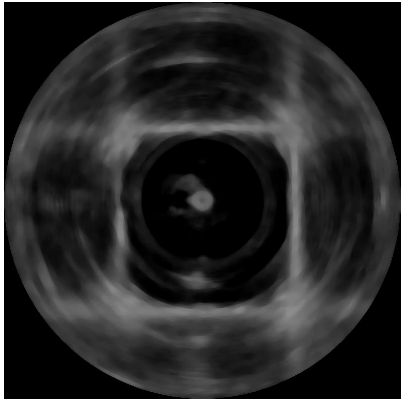


Fig 29. Median filter

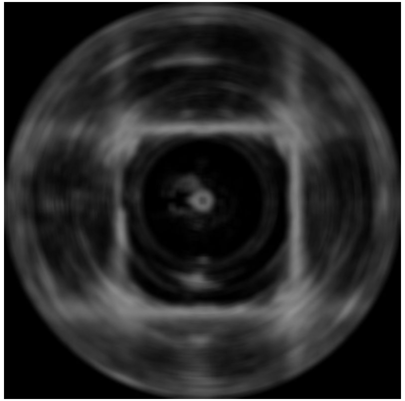


Fig 30. Mean filter

Obstacle Detection - Simple threshold

- Calculate variation of intensities
- Define a threshold
    - Variations above that threshold approach
- Increased blur provides best results

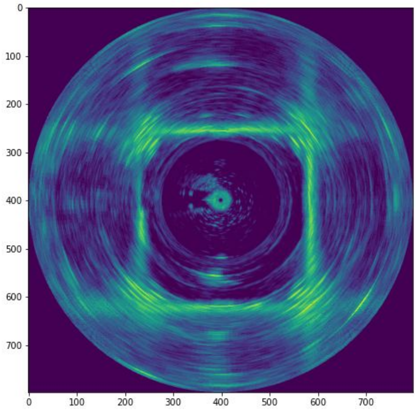


Fig 23. Original Image

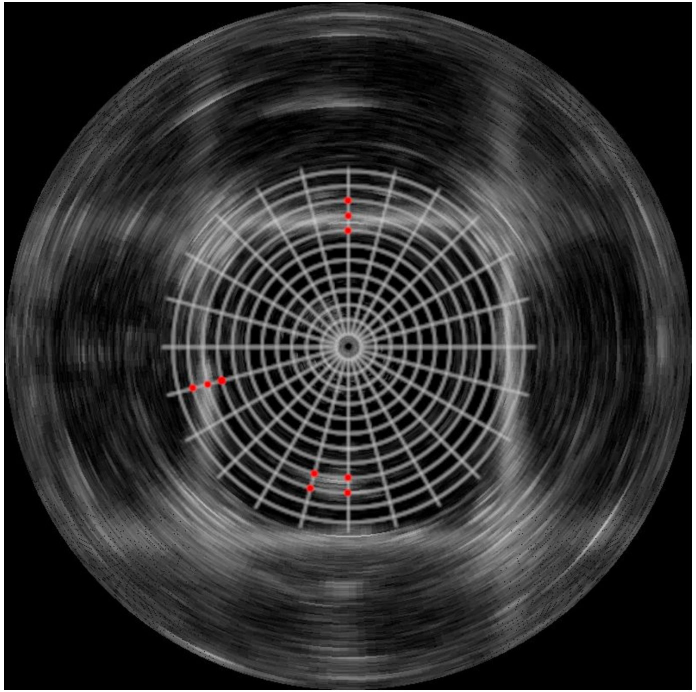


Fig 24. Scan with identified edges using simple threshold approach

Raycasting - Bresenham vs DDA

- Bresenham's line algorithm:
    - Travels in all axis at once
    - Handles diagonal transitions very well
    - Very accurate
- Digital Differential Analyzer (DDA)
    - Travels in only one axis at a time
    - Goes in the direction of the closest axis
    - May miss the target (see next slide)

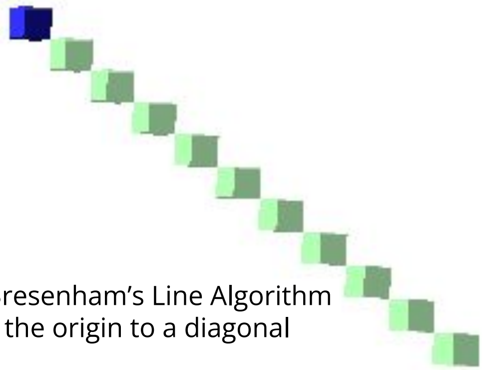


Fig 11. Bresenham's Line Algorithm from the origin to a diagonal

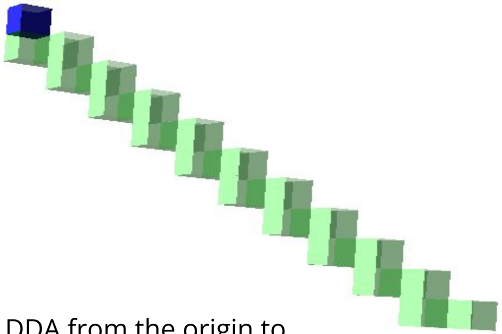


Fig 12. DDA from the origin to a diagonal

Octomaps/Octrees In Our Project

- The octrees used in the max depth 16;
  - This depth leads to a 8^16-1 = 2.8147498e+14 nodes.
- Resolution of 1 cm;
  - This resolution with the amount of nodes available lets up map a volume of 655 m^3.
- The **octomap** will be **probabilistic**:
  - 3 types of cells: free, occupied, and unknown;
  - Each cell has a probability of being empty (**log-odds**)
    - > 0 more likely to be occupied
    - < 0 more likely to be empty
  - Unknown cells are uninitialized

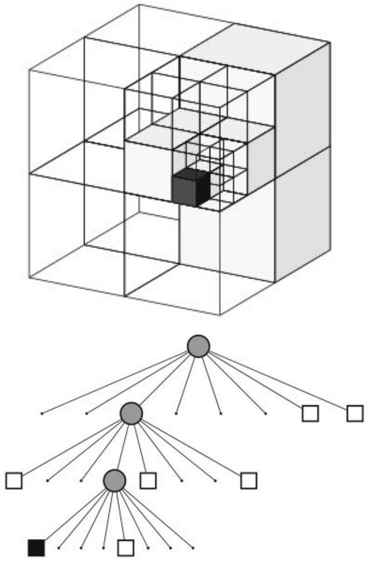


Fig 1. Example of octree storing occupied (black) and free (white) cells (Hornung et al.)

Probabilistic Mapping [1]

$$P(n|z_{1:t}) = \left[1 + \frac{1-P(n|z_t)}{P(n|z_t)} \frac{1-P(n|z_{1:t-1})}{P(n|z_{1:t-1})} \frac{P(n)}{1-P(n)}\right]^{-1}$$

Fig 11. Probability update formula.

- Which can be converted to log-odds notation:
    - More efficient – Reduces multiplications/divisions

$$L(n|z_{1:t}) = L(n|z_{1:t-1}) + L(n|z_t)$$

Fig 12. Probability update formula in log-odds.

$$L(n) = log(\frac{P(n)}{1-P(n)})$$

Fig 13. Log odds formula.

Octovis Exporting

- Exporting to octovis format;
- Allows visualization of results using the octovis tool;
- Implies some loss of information:
  - Node can only be full, empty, or unknown;
  - There’s no distinction between stable and unstable nodes.
- Tool works well for the current use case, but has multiple problems that may prove disadvantageous in the future:
  - Clipping artifacts in dense areas;
  - Bugs on the controls;
  - Incorrect object culling when zoomed in (it can be impossible to observe small objects when the zoom level is close to the maximum).